Exploiting Feature Covariance in High-Dimensional Online Learning

Justin Ma (UCSD → Berkeley), Alex Kulesza (UPenn),

Mark Dredze (Johns Hopkins), Koby Crammer (The Technion),
Lawrence K. Saul (UCSD), Fernando Pereira (Google)

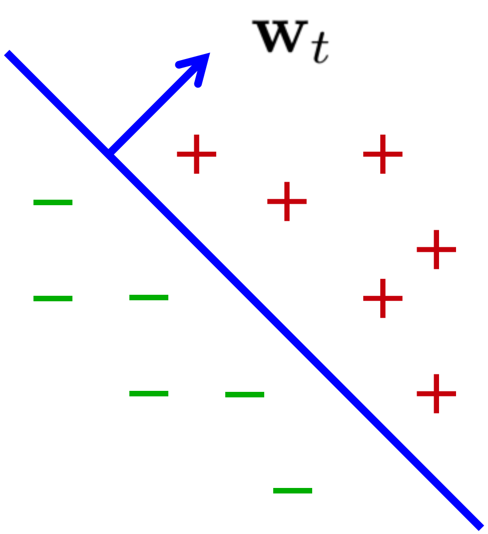
Presentation for AISTATS

May 14, 2010

UCSDCSE
Computer Science and Engineering

Online Learning of Linear Classifiers

- Input $\mathbf{x}_t$
- Predict $\hat{y}_t = \mathrm{sign}(\mathbf{w}_t \cdot \mathbf{x}_t)$
- Receive label $y_t \in \{-1, +1\}$
- Record error if $y_t \neq \hat{y}_t$
- Modify $\mathbf{w}_t$



Real-World Motivations

- Industrial-scale applications
  - Large data sets ($\sim 10^6$—10^9 examples)
  - Nonstationary data, drifting concepts

- Attractions of online learning
  - Single pass over data
  - Incremental update
  - Low overhead in storage & compute

UCSDCSE
Computer Science and Engineering

High-Dimensional Applications

- Ex: sentiment classification, malicious URL detection, Web spam
- Bag-of-words $\sim 10^6$ features
- New features introduced over time

Sentiment classification

Malicious URLs

Which online algorithm?

Perceptron [Rosenblatt, 1958]

Stochastic gradient [generalization in Bottou, 1998]
Bayesian logistic regression [MacKay, 1992] [Jaakkola & Jordan, 2000]

Online convex programming [Zinkevich, 2003]
Second-order perceptron [Cesa-Bianchi et al, 2005]
Passive-aggressive [Crammer et al, 2006]

Confidence-weighted [Dredze et al, 2008]
Online ellipsoid method [Yang et al, 2009]
AROW [Crammer et al, 2009]
AdaGrad [Duchi et al, 2010]

Trend toward more complex updates

…(e.g., using **2nd-order** information)

UCSDCSE Computer Science and Engineering

Which online algorithm?

Perceptron [Rosenblatt, 1958]

Stochastic gradient [generalization in Bottou, 1998]
Bayesian logistic regression [MacKay, 1992] [Jaakkola & Jordan, 2000]

Online convex programming [Zinkevich, 2003]
Second-order perceptron [Cesa-Bianchi et al, 2005]
Passive-aggressive [Crammer et al, 2006]

Confidence-weighted [Dredze et al, 2008]
Online ellipsoid method [Yang et al, 2009]
AROW [Crammer et al, 2009]
AdaGrad [Duchi et al, 2010]

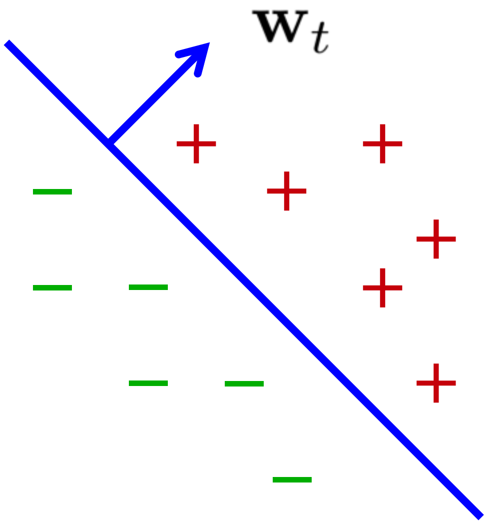
Trend toward more complex updates

…(e.g., using **2nd-order** information)

UCSDCSE
Computer Science and Engineering

Perceptron

[Rosenblatt, 1958]



- Per-mistake update:

$$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + y_t \mathbf{x}_t$$

- Convergence in finite rounds for separable data

Passive-Aggressive (PA) Algorithm

[Crammer et al., 2006]

- Constrained optimization

$$\mathbf{w}_{t+1} \leftarrow \operatorname*{argmin}_{\mathbf{w}} \quad \frac{1}{2}\|\mathbf{w}_t - \mathbf{w}\|^2$$
$$\text{s.t.} \quad y_t(\mathbf{w} \cdot \mathbf{x}_t) \geq 1$$

- Closed-form update

Amount of error

Proportional

$$\boxed{\alpha_t = \max\left\{\frac{1 - y_t(\mathbf{w}_t \cdot \mathbf{x}_t)}{\|\mathbf{x}_t\|^2}, 0\right\}}$$

$$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + \boxed{\alpha_t} y_t \mathbf{x}_t$$

UCSDCSE Computer Science and Engineering

Confidence-Weighted (CW) Learning

[Dredze et al., 2008] [Crammer et al., 2009]

- Gaussian distribution over weight vector:

$$\mathbf{w}_t \sim \mathcal{N}(\boldsymbol{\mu}_t, \boxed{\boldsymbol{\Sigma}_t^{-1}})$$

How to represent this matrix?

- Constrained problem:

$$(\boldsymbol{\mu}_{t+1}, \boldsymbol{\Sigma}_{t+1}^{-1}) \leftarrow \underset{\boldsymbol{\mu}, \boldsymbol{\Sigma}^{-1}}{\operatorname{argmin}} \ \mathrm{KL}(\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}^{-1}) \| \mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t^{-1}))$$
$$\text{s.t. } \Pr[y_t(\mathbf{w} \cdot \mathbf{x}_t) \geq 0] \geq \eta$$

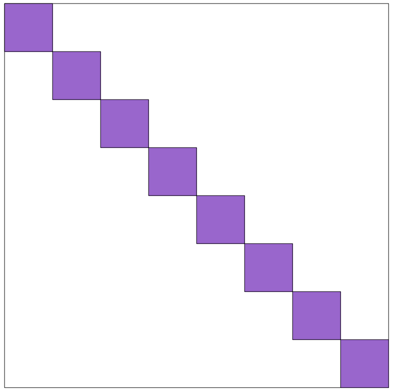
- Closed-form update:

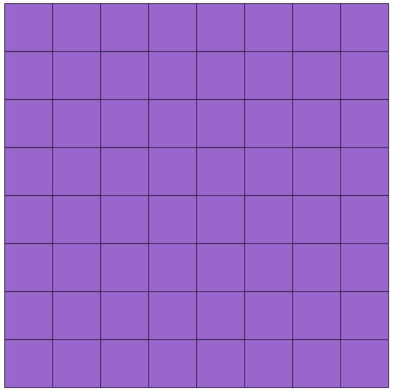
$$\boldsymbol{\mu}_{t+1} \leftarrow \boldsymbol{\mu}_t + \alpha_t y_t \boxed{\boldsymbol{\Sigma}_t \mathbf{x}_t}$$
$$\boldsymbol{\Sigma}_{t+1}^{-1} \leftarrow \boldsymbol{\Sigma}_t^{-1} + \frac{\alpha_t \phi}{\sqrt{u_t}} \mathbf{x}_t \mathbf{x}_t^\top$$

Update features at different rates

UCSDCSE Computer Science and Engineering

Representing Correlations

$\Sigma^{-1} =$ 

$\Sigma^{-1} =$ 

Diagonal

- ✔ $O(n)$ storage
- ✔ Low compute time

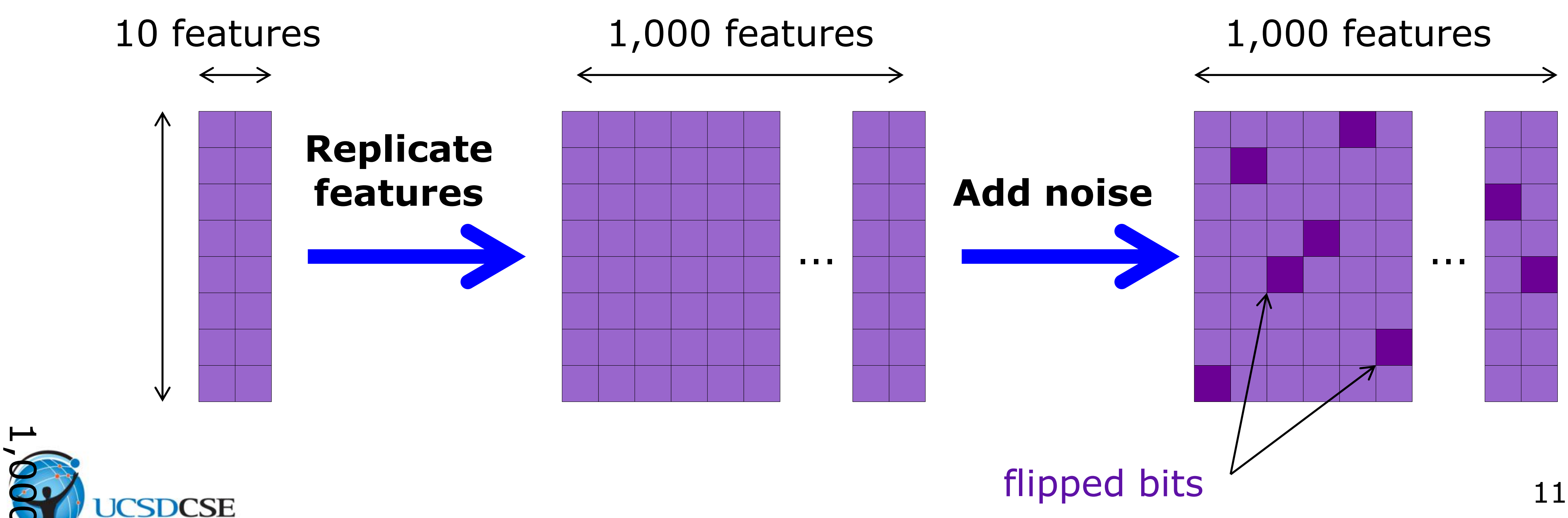
Full

- ✗ $O(n^2)$ storage
- ✗ High compute time

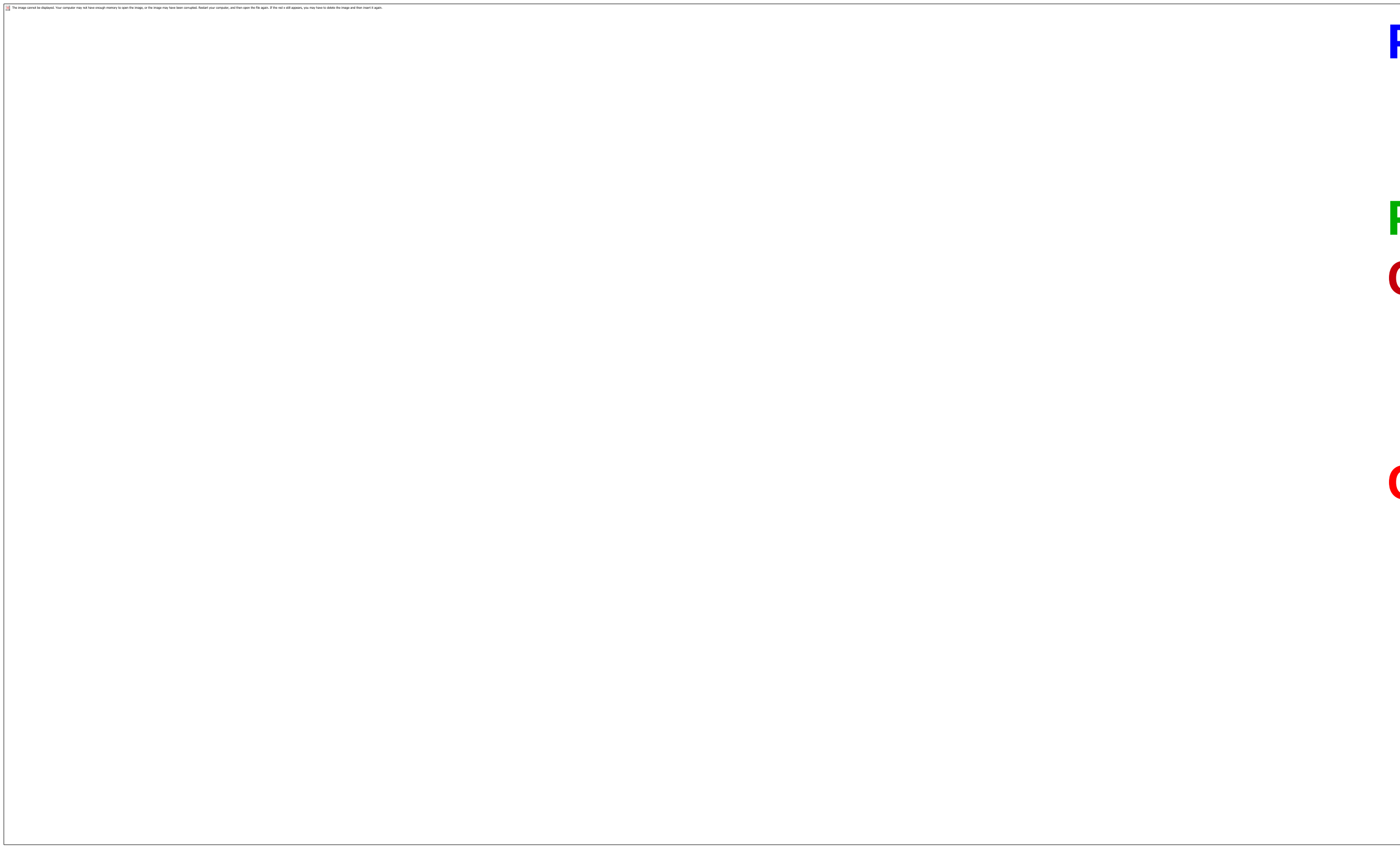
Why bother with full?

Benefits of Full

- When do we benefit from full covariance?
- Synthetic experiment: noisy correlated features
  - 100 runs, 1,000 examples, 1,000 binary features
  - 5% of features flipped



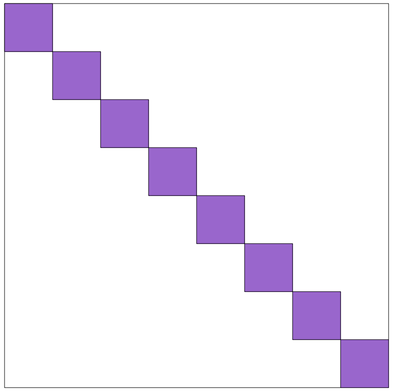
Noisy Correlated Features



- **Full** exploits correlations

Representing Correlations

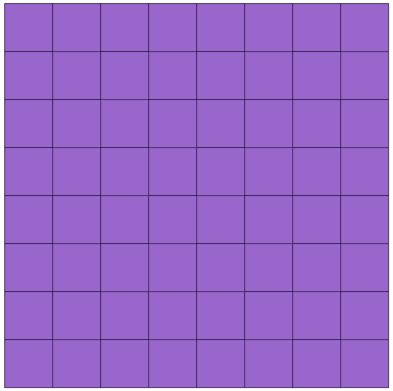
$\Sigma^{-1} =$



Diagonal

- ✔ O(n) storage
- ✔ Low compute time
- ✔ Ignores correlations

$\Sigma^{-1} =$



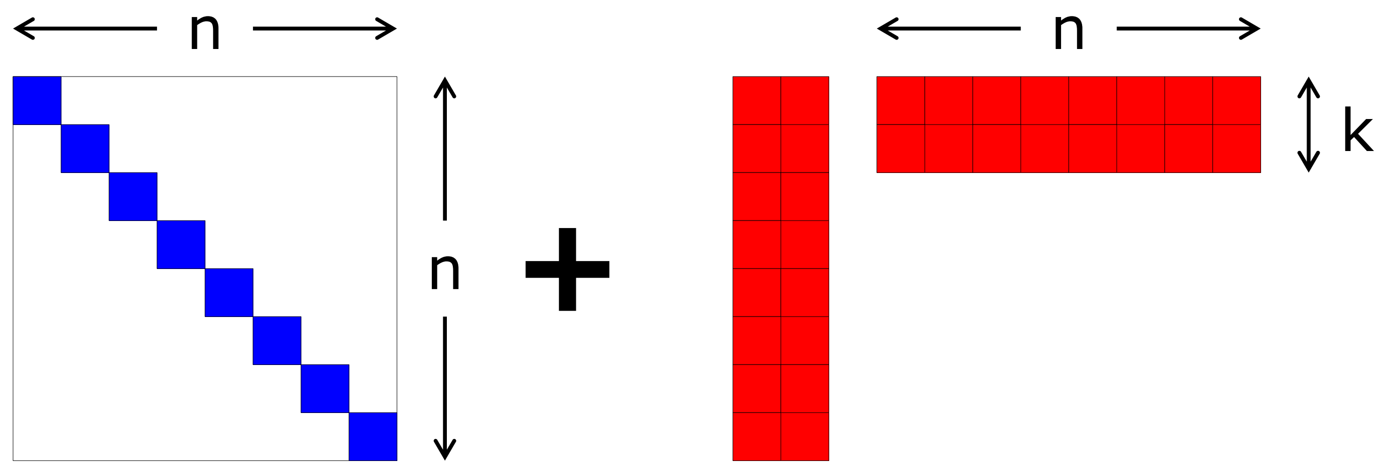
Full

- ✗ O(n^2) storage
- ✗ High compute time
- ✗ Exploits correlations

Can obtain benefits of both?

UCSDCSE
Computer Science and Engineering

Factored Approximation



$$\Sigma^{-1} = D + RR^T$$

- Approx. full inv. covariance (called precision)
- O(kn) storage (k = number of factors)
- Compress matrix updates – factor analysis

Factor Analysis

- Exact and approximate distributions

$$
\begin{aligned}
\rightarrow P(\mathbf{w}_t) &= \mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t^{-1}) \\
\rightarrow \widehat{P}(\mathbf{w}_t) &= \mathcal{N}(\boldsymbol{\mu}_t, \boxed{\boldsymbol{D}_t + \boldsymbol{R}_t \boldsymbol{R}_t^\top})
\end{aligned}
$$

- Minimize KL divergence using EM procedure

$$\min \ \mathrm{KL}(P_t, \widehat{P}_t)$$

- Computation cost: $O(nk^2)$

UCSDCSE
Computer Science and Engineering

Synthetic with CW-fact

Perceptron

PA

CW-diag

CW-fact4

CW-full

Cumulative r

- **Full** exploits correlations
- **Factored** approximates full
- **40x** less memory

Benefits of approximating full in real-world applications?

Detecting Malicious URLs

- Live feature collection of URLs
- Per trial: 200,000 examples, 10^6 features (mostly binary)

Additional mistakes
by CW-diag

Perceptron

PA

CW-diag
CW-fact16

Factored relative improvement: 5%

Web Spam Classification

- Web pages [PASCAL Large-Scale Learning competition]
- Per-run: 175,000 examples, 680,000 features (text 3-grams)

The image cannot be displayed. Your computer may not have enough memory to open the image, or the image may have been corrupted. Restart your computer, and then open the file again. If the red x still appears, you may have to delete the image and then insert it again.

Perceptron

PA

Additional mistakes by CW-diag

CW-diag

CW-fact16

Factored relative improvement: 18%

Document Classification

[Dredze et al., 2008]

- 2,000—18,000 examples, 10^4—10^6 features

The image cannot be displayed. Your computer may not have enough memory to open the image, or the image may have been corrupted. Restart your computer, and then open the file again. If the red x still appears, you may have to delete the image and then insert it again.

Improvement by CW-fact8

Reuters 20 Newsgroups Sentiment

Relative

UCSDCSE
Computer Science and Engineering

Conclusion

- Full and factored covariance help when...
  - features are correlated
  - #features > #examples
    (not needed when #examples > #features [Sec. 2])
- Factored improves high-dimensional apps
  - NLP, URL classification, Web spam, others
- Future work
  - Correlation modeling in other online algorithms?
  - Other ways to model correlation structure?

Code is Available

http://sysnet.ucsd.edu/projects/url/

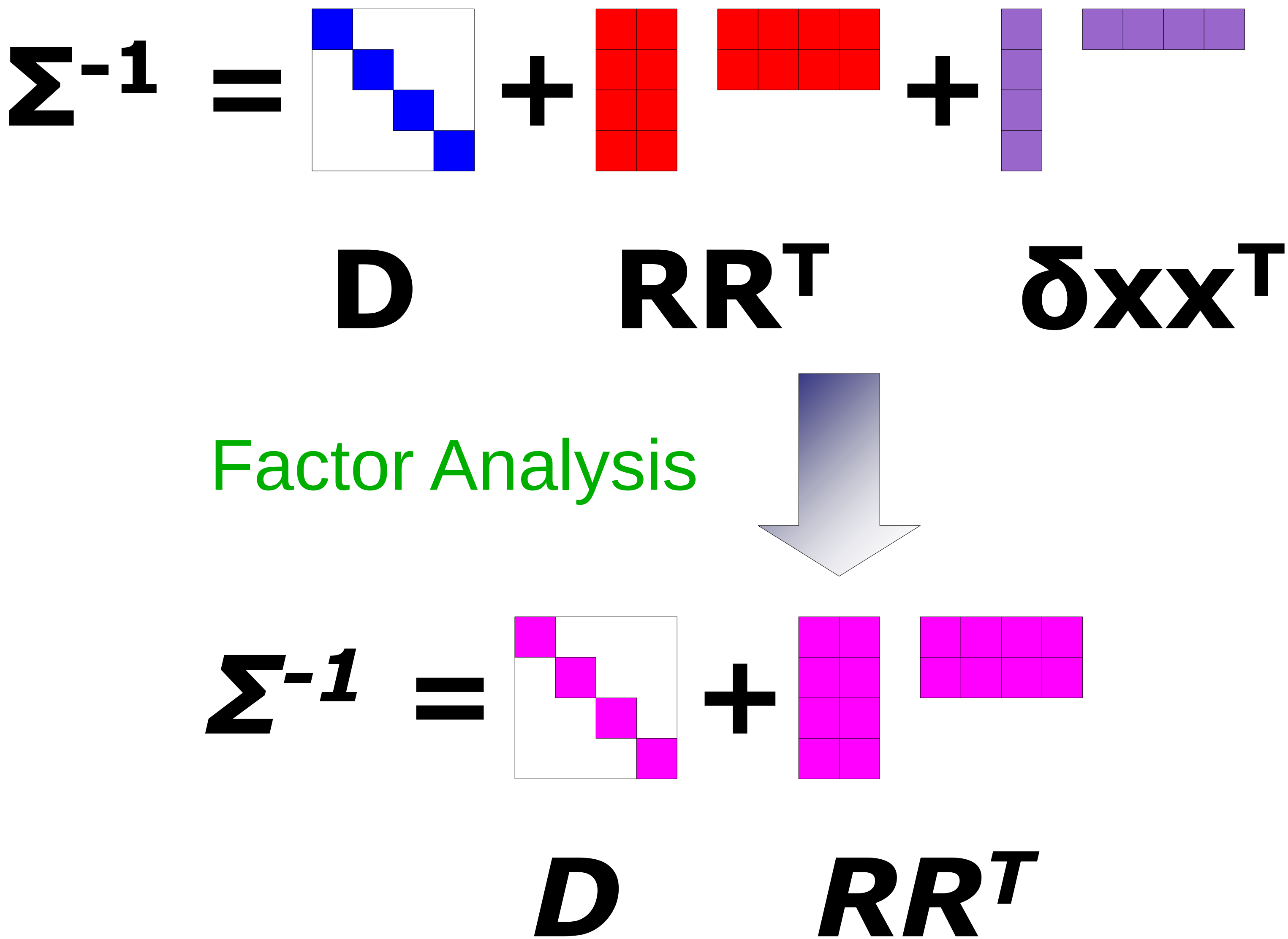
- Coauthors in Sardinia...

Alex

Koby

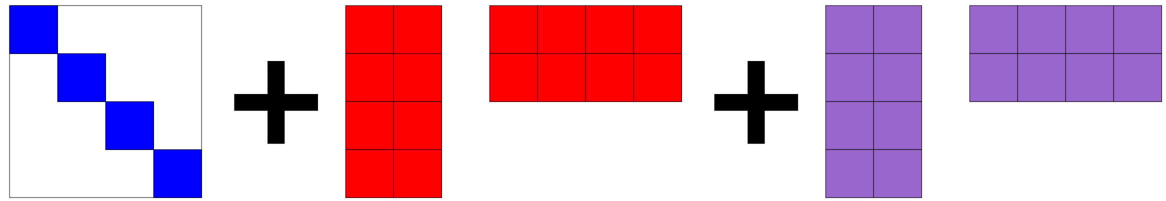
UCSDCSE Computer Science and Engineering

Update Compression



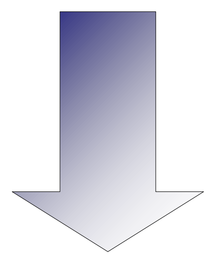
Buffering Updates

$\Sigma^{-1} =$

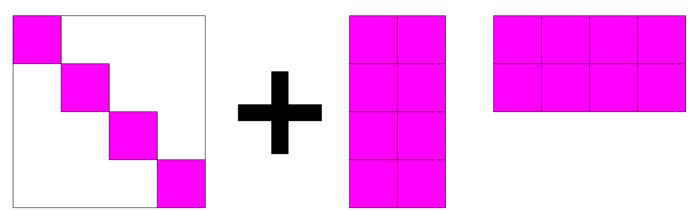


$D \qquad RR^T \qquad BB^T$

Factor Analysis



$\Sigma^{-1} =$



$D \qquad RR^T$